МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования «Южно-Уральский государственный университет»

(национальный исследовательский университет)

Высшая школа электроники и компьютерных наук

Кафедра системного программирования

Работа с файлами

(название практической работы)

ОТЧЕТ

по практической работе № 7

по дисциплине «Функциональное и логическое программирование»

Выполнил:

студент группы КЭ–401

/ Д.А. Иванов /

(подпись)

« » 2024 г.

Проверил:

преподаватель кафедры СП

/ М.А. Ческидова /

(подпись)

« » 2024 г.

Оглавление

[1. Задание 1 3](#_Toc179292138)

[2. Задание 2 5](#_Toc179292139)

[3. Задание 3 7](#_Toc179292140)

[4. Задание 4 9](#_Toc179292141)

[5. Задание 5 10](#_Toc179292142)

[6. Задание 6 11](#_Toc179292143)

[7. Задание 7 13](#_Toc179292144)

[8. ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ 14](#_Toc179292145)

# Задание 1

Написать программу формирования текстового файла. Строка считывается с клавиатуры.

На листинге 1 представлен код для формирования текстового файла.

Листинг 1 ­– Формирования текстового файла

% Предикат для записи строки в файл

write\_to\_file :-

% Считываем строку с клавиатуры

write('Введите строку: '),

read\_line\_to\_codes(user\_input, Input),

% Указываем имя файла

atom\_string(Filename, 'output.txt'),

% Открываем файл для записи

open(Filename, write, Stream),

% Записываем строку в файл

format(Stream, '~s', [Input]),

% Закрываем файл

close(Stream),

% Выводим сообщение об успешной записи

write('Строка успешно записана в файл output.txt.').

% Основной предикат для запуска программы

main :-

write\_to\_file.

% Запускаем программу

:- main.

На рисунке 1 представлены результаты выполнения этой программы.

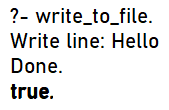


Рисунок 1 – Результат выполнения задания 1

# Задание 2

Используя стандартный предикат openappend добавить информацию в текстовый файл.

В SWI -Prolog отсутствует предикат openappend, вместо него используется предикат open с параметром append.

На листинге 2 представлен код программы для добавления текста в файл.

Листинг 2 – Добавление текста в файл

% Предикат для добавления строки в конец файла

append\_to\_file :-

% Считываем строку с клавиатуры

write('Write line: '),

read\_line\_to\_codes(user\_input, Input),

% Указываем имя файла

atom\_string(Filename, 'output.txt'),

% Открываем файл для добавления

open(Filename, append, Stream),

% Записываем строку в файл

format(Stream, '~s', [Input]),

% Закрываем файл

close(Stream),

% Выводим сообщение об успешной записи

write('Success Append.').

Результат представлен на рисунке 2.

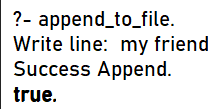


Рисунок 2 – Результат выполнения задания 2

В итоге выполнения двух первых заданий на выходе получился файл output.txt. Содержимое файла представлено на рисунке 3.

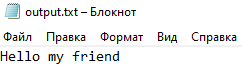


Рисунок 3 – Файл output.txt

# Задание 3

Пусть имеется файл на диске. Необходимо считать данные из файла для последующей обработки.

На листинге 3 представлен код программы для считывания данных из файла для последующей обработки.

Листинг 3 – Считывание данных из файла

% Предикат для чтения и обработки данных из файла

read\_and\_process\_file :-

% Указываем имя файла

atom\_string(Filename, 'input.txt'),

% Проверяем, существует ли файл

( exists\_file(Filename)

-> % Если файл существует, открываем его для чтения

open(Filename, read, Stream),

% Читаем и обрабатываем содержимое файла

read\_file\_contents(Stream),

% Закрываем файл

close(Stream)

; % Если файл не существует, выводим сообщение об ошибке

format('Error: File not exist: ~w', [Filename]), nl

).

% Предикат для чтения содержимого файла

read\_file\_contents(Stream) :-

% Читаем строку из файла

read\_line\_to\_string(Stream, Line),

% Проверяем, не достигнут ли конец файла

( Line \== end\_of\_file

-> % Если не конец файла, обрабатываем строку

process\_line(Line),

% Рекурсивно вызываем предикат для чтения следующей строки

read\_file\_contents(Stream)

; % Если достигнут конец файла, завершаем чтение

true

).

% Предикат для обработки одной строки

process\_line(Line) :-

% Последующая обработка текста

% В данном случае просто вывод текста на экран

format('Input Text:~n~w', [Line]).

На рисунке 4 представлен результат выполнения задания 3.

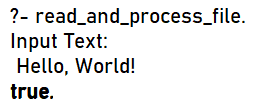


Рисунок 4 – Результат выполнения задания 3

# индивидуальное задание

Создайте предикат, вычисляющий сумму чисел, хранящихся в файле.

На листинге 4 представлен код программы для вычисления суммы чисел, хранящихся в файле.

Листинг 4 ­– Вычисление суммы чисел в файле

% Предикат для суммирования чисел в файле

sum\_numbers\_in\_file :-

% Указываем имя файла

atom\_string(Filename, 'numbers.txt'),

% Проверяем, существует ли файл

( exists\_file(Filename)

-> % Если файл существует, открываем его для чтения

open(Filename, read, Stream),

% Читаем и обрабатываем содержимое файла

sum\_lines(Stream, 0, Sum),

% Закрываем файл

close(Stream),

% Выводим результат

format('Sum: ~w', [Sum]), nl

; % Если файл не существует, выводим сообщение об ошибке

format('Error: File not exist: ~w', [Filename]), nl

).

% Предикат для чтения содержимого файла

sum\_lines(Stream, Acc, Sum) :-

% Читаем строку из файла

read\_line\_to\_string(Stream, Line),

% Проверяем, не достигнут ли конец файла

( Line \== end\_of\_file

-> % Если не конец файла, пытаемся преобразовать строку в число

( number\_string(Number, Line)

-> % Если строка представляет число, добавляем его к аккумулятору

NewAcc is Acc + Number,

% Рекурсивно вызываем предикат для чтения следующей строки

sum\_lines(Stream, NewAcc, Sum)

; % Если строка не представляет число, пропускаем её

sum\_lines(Stream, Acc, Sum)

)

; % Если достигнут конец файла, возвращаем аккумулятор как сумму

Sum = Acc

).

На рисунке 4 представлен результат выполнения этой программы.

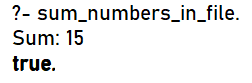


Рисунок 4 – Результат выполнения индивидуального задания